



[10191/3300]

AK  
W

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants : Michael BEUTEN et al.  
For : METHOD FOR DYNAMIC MEMORY MANAGEMENT  
Application Serial No. : 10/633,113  
Filed : August 1, 2003  
Examiner : Mardochee CHERY  
Group Art Unit : 2188  
Confirmation No. : 3639

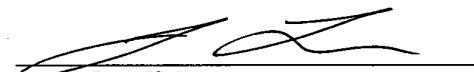
Mail Stop Appeal Brief-Patents  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on:

Date: January 8, 2008

Reg. No. 36,197

Signature:

  
Jong H. Lee

**APPELLANTS' REPLY BRIEF IN RESPONSE TO  
EXAMINER'S ANSWER (UNDER 37 C.F.R. § 41.41)**

S I R :

In response to the Examiner's Answer mailed on November 9, 2007 regarding the final rejection of claims 1-10 and 12 in the above-identified application, Applicants submit the following arguments in support of the appeal of the final rejection.

## ARGUMENT

The issue presented for review in this case is whether claims 1-10 and 12 are unpatentable under 35 U.S.C. § 103(a) over U.S. Patent No. 6,088,777 ("Sorber") in view of U.S. Patent No. 6,192,457 ("Porterfield") and U.S. Patent No. 6,141,756 ("Bright"). Applicants respectfully submit that the rejections should be reversed for at least the following reasons.

In order for a claim to be rejected for obviousness under 35 U.S.C. § 103(a), the prior art teach or suggest each element of the claim. See Northern Telecom, Inc. v. Datapoint Corp., 908 F.2d 931, 934 (Fed. Cir. 1990), cert. denied, 111 S. Ct. 296 (1990); In re Bond, 910 F.2d 831, 834 (Fed. Cir. 1990). To establish a *prima facie* case of obviousness, the Examiner must show, *inter alia*, that there is some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify or combine the references, and that, when so modified or combined, the prior art teaches or suggests all of the claim limitations. M.P.E.P. §2143. In addition, as clearly indicated by the Supreme Court, it is "important to identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the [prior art] elements" in the manner claimed. See KSR Int'l Co. v. Teleflex, Inc., 127 S. Ct. 1727 (2007). To the extent that the Examiner may be relying on the doctrine of inherent disclosure for the obviousness rejection, the Examiner must provide a "basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristics necessarily flow from the teachings of the applied art." (See M.P.E.P. § 2112; emphasis in original; see also Ex parte Levy, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990)).

Independent claim 1 recites, in relevant parts, "**providing a first memory block in the memory device; storing a startup program in the first memory block; providing additional memory blocks; and connecting the first memory block and the additional memory blocks by a chained list; wherein the memory device is checked before the chained list is executed and the startup program obtains data for a check from the additional memory blocks.**" Independent claims 6, 8 and 12 recite substantially similar features as the above-recited features of claim 1.

In support of the rejection of claim 1, the Examiner contends that Sorber teaches the following: “storing a **startup program in the first memory block** [col. 7, par. 2]; providing additional memory blocks [col. 3, par. 2]; and **connecting the first memory block and the additional memory blocks by a chained list** [col. 16, lines 54-57].” However, the actual disclosure of the cited sections of Sorber simply does not support the Examiner’s contentions. Among other things, one of the critical flaws in the Examiner’s contentions is that Sorber actually disclose that **the startup program is contained in the program memory 20, which is separately located from the data memory 22**, and any “linking” of blocks of memory mentioned in Sorber has to do with **allocation of available space within the data memory 22**, but there is no “linking” (by a chained list) of any **memory block storing a startup program** to another **memory block in data memory 22**, as explained in detail below.

Col. 3, par. 2 of Sorber (section cited by Examiner) discusses “dynamic allocation of memory” performed by operating system functions, which allocation involves address pointers, and col. 16, l. 54-57 (also cited by the Examiner) mentions that “each available memory block is linked to a next, available memory block in the same class and a first available memory block in the same class is indicated by a pointer.” However, Sorber clearly indicates that the “dynamic allocation” of memory space has to do with **allocation of available space within the data memory 22**: “the memory manager 26 determines from the operating system **how much of the memory 22 is available for dynamic allocation**, . . . and it is selectively dynamically allocated by the memory manager 26 to various requesting processes.” (Col. 7, l. 18-27). Regarding the management of the memory blocks **in memory 22**, Sorber indicates that **memory blocks** in the same class are linked together (col. 7, l. 29-32), and “[i]f the memory block is part of a message, it may be linked to other blocks in a list, (i.e., string of linked blocks), before the pointer to the first block in the message string is give to the next process.” (Col. 8, l. 55-59). Accordingly, any “linking” of blocks of memory mentioned in Sorber has to do with **allocation of available space within the data memory 22**, but there is no “linking” (by a chained list) of any **memory block storing a startup program** to another **memory block in data memory 22**. To the extent the Examiner may be implicitly contending that col. 7, l. 15-18 (“[w]hen the data processing and memory system 10 is booted-up an the operating system and other application software are

loaded”) somehow suggests that **a startup program** is loaded into the data memory 22, there is no indication in Sorber that a startup program is loaded into data memory 22. Even if one assumed for the sake of argument that a startup program is somehow loaded into data memory 22 (which is not disclosed in Sorber), it is clearly not inherent that a **memory block storing a startup program** is **linked by a chained list** to another **memory block in data memory 22**, particularly since Sorber discloses linking only those **memory blocks within the same class**, (col. 7, l. 29-32), and no linking of memory blocks of different classes is disclosed.

In the “Response to Argument” section of the Examiner’s Answer, the Examiner contends that Applicants are incorrect in asserting that “**Sorber explicitly indicates that the startup program is contained in the program memory 20, which is separately located from the data memory 22.**” In this regard, the Examiner contends “the ‘Data Processing and Memory System’ of Fig. 2 of Sorber is an **integrated circuit** built on a single chip having all the components therein interconnected and functioning as one ‘Data Processing and Memory System’ where all the circuitry and devices are all integrated into and work as one memory system as shown in Fig. 13.” However, it is patently obvious to a casual observer that **Fig. 2 is a functional block diagram**, and there is **absolutely no support** in Sorber that “the ‘Data Processing and Memory System’ of Fig. 2 of Sorber is **an integrated circuit built on a single chip . . . where all the circuitry and devices are all integrated into and work as one memory system.**” To the extent the Examiner may be implicitly contending that Fig. 2 **inherently** teaches an integrated circuit built on a single chip, this contention must fail because there is no “basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristics **necessarily flow** from the teachings of the applied art.” At best, the Examiner’s assertion is nothing more than a wishful conclusory statement that has no basis in fact. In any case, Fig. 2 clearly shows that program memory 20 is **connected by an external communications bus 18** to a **separately located data memory 22**, as described in col. 7, l. 1-14 of Sorber.

Independent of the above, to the extent the Examiner cites col. 7, l. 15-25 of Sorber as somehow negating the Applicants’ assertion that “Sorber explicitly indicates that the startup

program is contained in the program memory 20, which is separately located from the data memory 22,” there is nothing in col. 7, l. 15-25 of Sorber that contradicts the Applicants’ assertion, and there is clearly nothing in col. 7, l. 15-25 that supports the Examiner’s contention that “the ‘Data Processing and Memory System’ of Fig. 2 of Sorber is **an integrated circuit built on a single chip . . . where all the circuitry and devices are all integrated into and work as one memory system.**” More specifically, nothing in col. 7, l. 15-25 of Sorber even remotely alters the disclosure in the preceding section (col. 7, l. 1-14) that program memory 20 is connected by an external communications bus 18 to a separately located data memory 22.

In addition, to the extent the Examiner continues to cite col. 16, l. 54-57 of Sorber as teaching the claimed feature of “connecting the first memory block [containing the startup program] and the **additional memory blocks by a chained list,**” Applicants note that col. 16, l. 54-57 merely indicates that “each **available memory block** is linked to a next, **available memory block** in the **same class** and a first available memory block in the same class is indicated by a pointer,” and it is intuitively obvious by example that an “**available memory block**” cannot be a **memory block containing the startup program**. In any case, there is absolutely no teaching or suggestion in Sorber that the **memory block containing the startup program** is connected to the “available memory block” recited in col. 16, l. 54-57.

Independent of the above, regarding the feature that “the memory device is checked before the chained list is executed and the startup program obtains data for a check from the **additional memory blocks,**” the Examiner once again contends in the Examiner’s Answer that col. 4, l. 3-13 of Porterfield disclose that “the startup program obtains data for **a check** from the additional memory blocks.” In this regard, the Examiner further contends that “BIOS . . . gets **data for running any data check from the memory blocks where it is stored.**” However, this assertion is not only a vague and ambiguous statement (**where exactly are the memory blocks storing the data for the check?**), but this assertion clearly does not address the actual claim limitations, i.e., the “startup program” in **the first memory block** of the memory device **obtains data for a check of the memory device from the additional memory blocks** of the memory device. The cited section (col. 4, lines 3-13) of Porterfield clearly does not disclose that

the “startup program” in the first memory bock of the memory device obtains data for a check of the memory device from the additional memory blocks of the same memory device; instead, this cited section of Porterfield merely discloses a system allocation table specifying system addresses for the various components of the computer system, and that the “addresses allocated for each computer device in the system address allocation table typically will be set by the Basic Input-Output System (BIOS) software when the computer system 50 is initialized upon being turned ON.” To the extent the Examiner may contending that a BIOS function would inherently satisfy the claimed feature that the “startup program” in the first memory bock of the memory device obtains data for a check of the memory device from the additional memory blocks of the memory device, there is no “basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristics **necessarily flow** from the teachings of the applied art.” Even if one assumes for the sake of argument that BIOS tests hardware at startup, this assumption simply does not provide any teaching or suggestion regarding **where the startup program (which is stored in the first memory block) obtains data for the check,** let alone provide any suggestion that **the startup program stored in the first memory block obtains data for a check of the memory device from the additional memory blocks.**

Independent of the above, regarding the feature that “**the memory device** [containing the startup program and the additional memory blocks] **is checked before the chained list is executed,**” the Examiner continues to insist that Bright discloses this feature in col. 3, l. 15-27. However, it is absolutely clear that there is no suggestion in Bright regarding checking of any memory device before any chained list is executed; instead, the cited section of Bright merely indicates use of encryption/decryption in downloading a program from an external device 103 by a separate processor 101. In order to overcome this glaring deficiency, the Examiner provides a **completely unsupported**, conclusory statement that “checksum and hash are used to implement executable programs using linked/chained lists.” However, not only is this assertion by the Examiner completely unsupported, but even if one assumes for the sake of argument that “checksum and hash” **may be** used to implement executable programs using linked/chained lists,” with which assumption Applicants do not agree, it is clear that there is no inherent reason why a linked/chained list would **have to be** implicated in connection with the teachings of Bright. Furthermore, even if one assumes for the sake of argument that a linked/chained list is

somehow implicated by the teachings of Bright, there still would not be any inherent reason why a **memory device containing the startup program and the additional memory blocks** would **have to be checked before any chained list is executed.**

Independent of the above, the Examiner has not provided any coherent reason why any person of ordinary skill in the art would be motivated to make the modifications asserted by the Examiner, particularly when one considers the completely different technologies involved in the applied references. For example, the teachings of Bright involve the use of encryption/decryption in downloading a program from an external device 103 by a separate processor 101, while the teachings of Porterfield involve implementing a graphics address remapping table as a virtual register in system memory, and the teachings of Sorber involve a memory management scheme for dynamic memory allocation corresponding to various data sizes. To the extent the Examiner appears to contend that the motivation to combine the applied references is merely that each of these references involve a memory-related operation, this alleged motivation clearly falls well short of the coherent rationale mandated by the Supreme Court in KSR Int'l Co. v. Teleflex, Inc..

For at least the foregoing reasons, the overall teachings of Sorber, Porterfield and Bright cannot possibly render independent claims 1, 6, 8 and 12, as well as their dependent claims 2-5, 7, 9 and 10, obvious.

**CONCLUSION**

For the preceding reasons, it is respectfully submitted that the rejections of claims 1-10 and 12 under 35 U.S.C. § 103(a) should be reversed.

Respectfully submitted,

 (R. No. 36,197)

Dated: January 8, 2008

By: JONG LEE for Gerard Messina  
Gerard A. Messina (Reg. No. 35,952)  
KENYON & KENYON LLP  
One Broadway  
New York, NY 10004  
(212) 425-7200  
**CUSTOMER NO. 26646**